

Roadmap Enhanced Improvement to the VSIMM Tracker via a Constrained Stochastic Context Free Grammar

Sijia Gao and Vikram Krishnamurthy*, *Fellow, IEEE*

Abstract—The aim of syntactic tracking is to classify spatio-temporal patterns of a target’s motion using natural language processing models. In this paper, we generalize earlier work by considering a constrained stochastic context free grammar (CSCFG) for modeling patterns confined to a roadmap. The constrained grammar facilitates modeling specific directions and road names in a roadmap. We present a novel particle filtering algorithm that exploits the CSCFG model for estimating the target’s patterns. This meta-level algorithm operates in conjunction with a base-level tracking algorithm. Extensive numerical results using simulated ground moving target indicator (GMTI) radar measurements show substantial improvement in target tracking accuracy.

I. INTRODUCTION

Consider a moving target confined to the road network illustrated in Fig. 1. Assume that the target is being tracked by a ground moving target indicator (GMTI) radar system. At each discrete time k , let \mathbf{z}_k denote the noisy GMTI measurement and \mathbf{x}_k denote the state vector comprising position and velocity of the target as it moves in two dimensional space. Also, d_k and l_k denote, respectively, the direction of motion and the location (road or intersection names) of the target.

Classical (base-level) tracking algorithms have been well studied in the literature [1] [2]. These include the variable structure interacting multiple model (VSIMM) tracker. In a VSIMM tracker, the direction sequence $d_{1:k} = (d_1, \dots, d_k)$ is modeled as a Markov chain with state space dependent on the location sequence $l_{1:k} = (l_1, \dots, l_k)$. These direction and location sequences are chosen so that the target is confined to roads and intersections in a roadmap.

At a higher level of abstraction (lower degree of spatial resolution and slower time scale), a moving target confined to a roadmap can be characterized by an ordered sequence of intersection names it traverses. For example, $v_0 v_2 v_7$ in Fig. 1 indicates a target which starts at intersection v_0 , traverses through v_2 and ends at v_7 . For convenience, we call an ordered sequence of intersection names as a *pattern*. In this paper, we consider a “syntactic” enhancement to the basic VSIMM setup. This syntactic enhancement operates at a higher (meta) level and models the pattern of a target. Our aim is to estimate the target’s pattern r^* such that

$$r^* = \underset{\forall r \in M}{\operatorname{argmax}} p_\theta(r | \mathbf{z}_{1:k})$$

S. Gao is with the Department of ECE, University of British Columbia, Vancouver, Canada. Vikram Krishnamurthy is with the Department of Electrical and Computer Engineering, Cornell University, N.Y., USA. vikramk@cornell.edu and is the corresponding author.

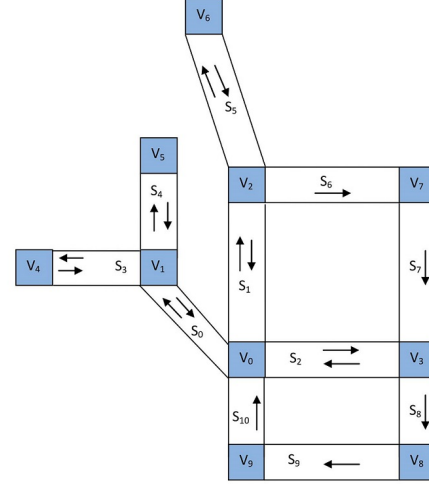


Figure 1. Example of a roadmap. s_0, \dots, s_{10} denote roads and v_0, \dots, v_9 denote intersections. Arrows depict the direction of a road (one way or two way road).

Here, $\mathbf{z}_{1:k} = (\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_k)$ is the noisy observation sequence recorded by a GMTI radar. r is a pattern, M denotes a set of patterns that a radar operator is interested in and θ denotes the roadmap statistic obtained from traffic data. Once the pattern r^* is estimated, it can be used as additional information to estimate the state vector \mathbf{x}_k of the target by computing the posterior $p(\mathbf{x}_k | \mathbf{z}_{1:k}, r^*)$.

The syntactic tracker we propose in this paper consists of two parts: a meta-level tracker and a base-level tracker. The meta-level tracker uses tracklets generated by the base-level tracker to model higher level patterns. The architecture of a syntactic tracker is illustrated in Fig. 2. Our key idea is to model the pattern of a target moving on a roadmap using a *constrained stochastic context free grammar* (CSCFG) on a weighted, directed graph. We then propose a novel particle filtering algorithm that combines the functionalities of CSCFG and VSIMM. Detailed numerical simulations show that the resulting tracker yields substantially improved estimates compared to a baseline tracker.

A. Syntactic Tracker Architectures

To give insight into the main ideas of this paper, we briefly describe two architectures of the syntactic trackers, namely (i) the mode based syntactic tracker proposed in this paper and

- In the stochastic case [6]:

$$\begin{aligned} \text{Markov chain} &\subset \text{Hidden Markov chain} \equiv \\ \text{Stochastic Regular Grammar} &\subset \text{SCFG} \subset \text{CSCFG} \end{aligned} \quad (2)$$

From a signal processing point of view, this paper deals with Bayesian signal processing algorithms for CSCFGs; the key application being syntactic target tracking. Put simply, at a meta-level, *we view the spatial trajectory of a target as a sequence of noisy alphabets generated by a CSCFG language*.¹ It turns out that Bayesian estimation algorithms (Earley Stolcke parser) for CSCFGs have polynomial computational cost (in the data length). Therefore, in the context of target tracking, we are able to derive a Rao-Blackwellized particle filter which uses the Earley Stolcke parser for estimating the posterior distribution of a CSCFG.

Why syntactic models for target tracking? As described in [4], [5], syntactic models arising in natural language processing such as SCFG and CSCFG are suitable for meta-level tracking since they form *generative models* for complex spatial trajectories of a target. For example, a SCFG is a generative model for a trajectory sequence $a^n b^n$ for an integer valued random variable n ; this trajectory models a target moving n steps in direction a followed by n steps in direction b . A Markov chain cannot exclusively generate such trajectories and is therefore not a generative model. Also there are polynomial computational cost (in the data length) Bayesian signal processing parsing algorithms for SCFGs and CSCFGs which make such models practical from an engineering point of view.

The main results of this paper are as follows: Sec. II and Sec. III discuss a 3-level model for the roadmap syntactic tracking problem. In Sec. IV, a novel CSCFG-driven particle filtering algorithm is given for the mode based syntactic tracker. Sec. V describes two CSCFG based models on a square grid. In Sec. VI, we compare numerical results between the mode based syntactic tracker (this paper) and the baseline VSIMM tracker. For the CSCFG based round trip model on the square grid, we compare performance between the CSCFG Viterbi tracker and the hidden Markov model (HMM) Viterbi tracker.

II. ROADMAP CONSTRAINED SYNTACTIC TRACKING: A 3-LEVEL MODEL

In this section, we construct a model for the roadmap constrained syntactic tracking problem. The model we propose operates at three levels of abstraction. At the highest level, we have the roadmap which is modeled as a directed, weighted graph. At the second level, we model the pattern (an ordered sequence of intersection names) of a target on the roadmap as a CSCFG. Finally at the lowest level, the mode sequence $q_{1:k}$ drives the base-level VSIMM state space model which has measurements from a GMTI radar.

¹In simple terms, a CSCFG permits both tree dependencies and serial dependencies. For an illustrative example that shows the difference between a Markov chain, SCFG and CSCFG, please see Sec. III-A and Fig. 6.

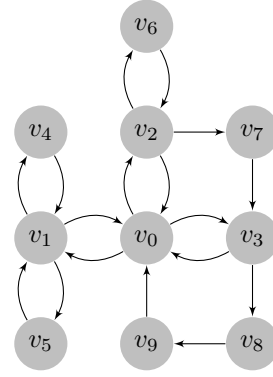


Figure 5. Formulation of the roadmap in Fig. 1 as a directed, weighted graph $\mathcal{G} = \{V, E, L\}$. v_0, \dots, v_9 denote road intersections.

A. Level 1: Roadmap as a Directed Weighted Graph

Our aim is to perform meta-level tracking of a target moving confined to a roadmap. We model the roadmap as a directed, weighted graph \mathcal{G} with vertices V , edges E and weights L

$$\mathcal{G} = \{V, E, L\}$$

The set of vertices $V = \{v_1, v_2, \dots, v_n\}$ denotes the n road intersections. The set of edges $E = \{e_{v_i v_j} | v_i, v_j \in V\}$ denotes the roads on the roadmap. The set of weights $L = \{l(e_{v_i v_j}) | e_{v_i v_j} \in E\}$ denotes lengths of the roads. Define a function θ on both vertices V and edges E . $\theta(e_{v_i v_j}), \forall e_{v_i v_j} \in E$ denotes the angle of road $e_{v_i v_j}$ with respect to a reference coordinate and $\theta(v_m), \forall v_m \in V$ denotes the set of angles of roads that intersect at the vertex v_m . The directed, weighted graph for the roadmap in Fig. 1 is presented in Fig. 5.

B. Level 2: CSCFG Model on Directed Weighted Graph for Targets Constrained to a Roadmap

The second level of the syntactic tracking model is a CSCFG model that ensures the target is confined to the roadmap graph $\mathcal{G} = \{V, E, L\}$. The model determines the target's mode sequence $q_{1:k}$ where

$$q_k = \{d_k, l_k\} \quad \text{with} \quad (3)$$

$$d_k \in \theta(l_k), l_k \in V \cup E$$

Here, d_k is the direction of motion of the target at time k . l_k denotes the edge (road) or vertex (intersection) where the target is located on \mathcal{G} . $\theta(\cdot)$ is defined in (1). A CSCFG is a 5-tuple of the form

$$\text{CSCFG} = \{\mathcal{N}, \mathcal{T}, S, \mathcal{R}, \mathcal{P}\}$$

where \mathcal{N} is a finite set of nonterminals and \mathcal{T} is a finite set of terminals (also called the alphabet) such that $\mathcal{N} \cap \mathcal{T} = \emptyset$. $S \in \mathcal{N}$ is chosen to be the start symbol. \mathcal{R} is a set of production rules of the form

$$\begin{aligned} X &\rightarrow \lambda | a, \quad X \in \mathcal{N}, X \neq S, a \in \mathcal{T}, \lambda \in \{\mathcal{N} \cup \mathcal{T}\}^* \\ S &\rightarrow \lambda \end{aligned}$$

which indicates the nonterminal X can be replaced with λ if the previous terminal is a . $\mathcal{P} : \mathcal{R} \rightarrow [0, 1]$ is a probability

function over production rules in \mathcal{R} such that

$$\sum_{n_{Xa}} P(X \rightarrow \lambda|a) = 1$$

$$\sum_{n_S} P(S \rightarrow \lambda) = 1$$

Here, n_{Xa} is the number of rules in \mathcal{R} associated with the nonterminal X and the terminal a . n_S is the number of rules that have the start symbol S on the left side of the arrow. Starting from S , replace the leftmost nonterminal (such deviations can be represented as a parse tree [8]) according to the production rules in \mathcal{R} and probabilities in \mathcal{P} , the output of a CSCFG is a string of terminals. A parsing algorithm for the CSCFG is illustrated in Appendix A.

Types of Target and Traffic Models: For illustrative purposes, we consider two types of target and traffic models for syntactic tracking. Let e_k denote the speed of a target at time k .

- 1) *Constant Traffic Flow or Constant Speed Model:* Assume that the traffic flow c throughout the road network is a positive random variable. Then the time taken to traverse a road segment of length $l(e_{v_i v_j})$ with speed limit ζ_i is $l(e_{v_i v_j})/(\zeta_i - c)$.

Alternatively, if the speed on a road or an intersection is an unknown constant modeled by a random variable c , then we have

$$e_k = c \quad (4)$$

Constant speed targets can model (approximately) pedestrians and bicycles.

- 2) *Average Speed Model:* Here the speed on a road at time k equals the speed averaged over all vehicular traffic moving on that road. Therefore, the speed on a road of vehicular traffic is location based (different roads) and time varying (peak or non-peak hours). Speed at an intersection is a known constant (denoted by c_2). Then we have

$$e_k = \begin{cases} \pi_E(l_k, k) & l_k \in E \\ c_2 & l_k \in V \end{cases} \quad (5)$$

where l_k is defined in (3). $\pi_E(l_k, k)$ denotes the speed averaged over all vehicular traffic moving on l_k at time k .

C. Level 3: GMTI Base Level Model

Here, we describe the third and final component of our 3-level tracking model. We construct a VSIMM model for the base-level target's kinematics²

which are measured by a GMTI radar system.

The target's state evolves as

$$\mathbf{x}_k = f(\mathbf{x}_{k-1}, d_{k-1:k}, e_{k-1:k}) + \tilde{\mathbf{w}}_k(d_k) \quad (6)$$

$$l_k = B(\mathbf{x}_k), d_k \in \theta(l_k)$$

Here, $\mathbf{x}_k = [x_k, y_k, \dot{x}_k, \dot{y}_k]'$ is the 4-dimensional state vector of the target at time k that comprises position and velocity components in the x and y directions. d_k and l_k are defined

in (3). e_k is the speed of the target specified in (4) and (5). $\theta(\cdot)$ is defined in (1). B is the function that maps \mathbf{x}_k to l_k . f is a nonlinear function and models the target's state process:

$$f = \begin{cases} f_0 & \text{if } d_k = d_{k-1}, e_k = e_{k-1} \\ f_1 & \text{otherwise} \end{cases} \quad (7)$$

where

$$f_0 = F\mathbf{x}_{k-1}$$

F is the state matrix defined by

$$F = \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

where T is the interval between GMTI measurements. f_1 in (7) denotes transforming the velocity components of $F\mathbf{x}_{k-1}$ to $e_k \cos d_k$ in the x direction and $e_k \sin d_k$ in the y direction. The state noise $\tilde{\mathbf{w}}_k(d_k)$ in (6) is a zero-mean white Gaussian process with covariance matrix $\tilde{Q}_k(d_k)$ computed as

$$\tilde{Q}_k(d_k) = GQ_k(d_k)G'$$

$$G = \begin{bmatrix} \frac{T^2}{2} & 0 \\ 0 & \frac{T^2}{2} \\ T & 0 \\ 0 & T \end{bmatrix}, \quad Q_k(d_k) = \rho_{d_k} \begin{bmatrix} \sigma_o^2 & 0 \\ 0 & \sigma_a^2 \end{bmatrix} \rho'_{d_k} \quad \text{with}$$

$$\rho_{d_k} = \begin{bmatrix} \sin d_k & \cos d_k \\ -\cos d_k & \sin d_k \end{bmatrix}$$

where $'$ denotes transpose, σ_o^2 is the variance along the direction of motion indicated by d_k and σ_a^2 is the variance along the direction of motion orthogonal to d_k .

The observation equation is specified by the GMTI radar:

$$\mathbf{z}_k = h(\mathbf{x}_k, \mathbf{c}_k) + \mathbf{v}_k \quad \text{where}$$

$$h(\mathbf{x}_k, \mathbf{c}_k) = \begin{bmatrix} \sqrt{(x_k - x_k^c)^2 + (y_k - y_k^c)^2 + (z_k^c)^2} \\ \frac{(\dot{x}_k - \dot{x}_k^c)(x_k - x_k^c) + (\dot{y}_k - \dot{y}_k^c)(y_k - y_k^c)}{\sqrt{(x_k - x_k^c)^2 + (y_k - y_k^c)^2 + (z_k^c)^2}} \\ \frac{180}{\pi} \text{atan2}(y_k - y_k^c, x_k - x_k^c) \end{bmatrix} \quad (8)$$

Here, $\mathbf{z}_k = [r_k, \dot{r}_k, \theta_k]'$ denotes the 3-dimensional noisy observation vector recorded by a GMTI radar at time k . r_k , \dot{r}_k , θ_k denote, respectively, the range, range rate and azimuth (in degrees, $(-180^\circ, 180^\circ)$). $\mathbf{c}_k = [x_k^c, y_k^c, \dot{x}_k^c, \dot{y}_k^c]'$ is the 4-dimensional state vector for the phase center of the GMTI radar's antenna on the aircraft it is mounted on. It comprises position and velocity components in the x and y directions. z_k^c is the (constant) altitude of the aircraft and the (constant) altitude of the target is assumed to be zero. atan2 denotes the four-quadrant inverse tangent (in radians). The observation noise \mathbf{v}_k in (8) is assumed to be a zero-mean white Gaussian process with covariance matrix

$$R = \begin{bmatrix} \sigma_{r_k}^2 & 0 & 0 \\ 0 & \sigma_{\dot{r}_k}^2 & 0 \\ 0 & 0 & \sigma_{\theta_k}^2 \end{bmatrix}$$

where σ_{r_k} , $\sigma_{\dot{r}_k}$ and σ_{θ_k} are standard deviations for range, range rate and azimuth, respectively. Note that R is a diagonal

²Our setup assumes a single target with no missing measurements or data association errors. Actually, missing measurements are easily handled at both the syntactic and base-level trackers.

matrix reflecting the assumption that the errors in the range, range rate and azimuth are uncorrelated.

III. EXAMPLES OF PATTERNS

Given the three level model described in Sec. II, we now elaborate on the Level 2 CSCFG model described in Sec. II-B. In particular, we discuss the two examples of constant speed targets and vehicular traffic in more detail.

First we need to define what a pattern is; recall that a pattern was defined informally in Sec. I as an ordered sequence of road intersection names. More formally, given the roadmap graph $\mathcal{G} = \{V, E, L\}$ formulated in Sec. II-A, a pattern is characterized by the sequence

$$\begin{aligned} r_{0:n} &= (r_0, \dots, r_n) \\ \text{where } r_i &\in V \quad i = 0, 1, \dots, n \\ e_{r_i r_{i+1}} &\in E \quad i = 0, 1, \dots, n-1 \end{aligned} \quad (9)$$

We assume that apart from kinematic measurements obtained by the radar, the type of a target is also known. In Sec. III-B and Sec. III-C, we build CSCFG based models for constant speed targets and vehicular traffic, respectively.

A. A Simple Illustrative Example

To give some intuition about the difference between a Markov chain, SCFG used in earlier work, and a CSCFG used in this paper (recall (2)), consider the following example.

Markov Chain. Construct a first-order Markov chain with trajectory a_1, \dots, a_n for some fixed time n . The dependency structure, which is a chain graph, is shown in Fig. 6(a).

SCFG and Arc Trajectory. An arc is an example of trajectory with a SCFG generative model.

- 1) Generate m, n as random positive integers with a pre-specified distribution.
- 2) Then generate the following three iid finite state sequences $a_1, \dots, a_m, b_1, \dots, b_n$ and c_1, \dots, c_m with specified probabilities. Concatenate these into a single string.

The dependency structure, which is a tree graph, is shown in Fig. 6(b). It can be verified via a pumping lemma [7] that a Markov chain is not a generative model for an arc since the integer m has an arbitrary (random) value.

CSCFG. A CSCFG trajectory has more general dependencies as follows.

- 1) Generate m, n as random positive integers with some pre-specified distribution.
- 2) Then generate the following three Markovian finite state sequences $a_1, \dots, a_m, b_1, \dots, b_n$ and c_1, \dots, c_m with specified transition probabilities. Concatenate these into a single string.

The dependency structure, which is a tree-chain graph, is shown in Fig. 6(c). The main point is that a CSCFG model facilitates both serial and tree dependencies. We refer to [6] for a detailed discussion of CSCFGs. In the remainder of this section we describe more sophisticated examples involving constant traffic flow and average speed models.

B. Constant Traffic Flow or Constant Speed Model

Here, we provide more details of the constant traffic flow or constant speed model in Sec. II-B. Given a pattern $r_{0:n}$, let $m_{r_i r_{i+1}}, i = 0, 1, \dots, n-1$, denote the sequence of vectors comprising directions and locations of a target as it traverses from vertex (intersection) r_i to r_{i+1} .

To provide a concrete example, consider the setup in Fig. 7(a) where a target moves from r_i to r_{i+1} . It moves in a constant direction $\theta(e_{r_i r_{i+1}})$ and via the locations $r_i, e_{r_i r_{i+1}}, r_{i+1}$. Therefore,

$$m_{r_i r_{i+1}} = \{\theta(e_{r_i r_{i+1}}), r_i\}^{i_1} \{\theta(e_{r_i r_{i+1}}), e_{r_i r_{i+1}}\}^{i_2} \{\theta(e_{r_i r_{i+1}}), r_{i+1}\}^{i_3} \quad (10)$$

where i_1, i_2, i_3 denote positive integers. In words: the target moving from r_i to r_{i+1} generates vectors $\{\theta(e_{r_i r_{i+1}}), r_i\}$ for i_1 time instants, followed by generating vectors $\{\theta(e_{r_i r_{i+1}}), e_{r_i r_{i+1}}\}$ for i_2 time instants, finally followed by generating vectors $\{\theta(e_{r_i r_{i+1}}), r_{i+1}\}$ for i_3 time instants. Therefore the total amount of time taken for traversing from r_i to r_{i+1} is

$$|m_{r_i r_{i+1}}| = i_1 + i_2 + i_3$$

$\mathcal{L}_{r_{0:n}}^{cs}$ is denoted as the sequence of vectors comprising directions and locations of a constant speed target as it traverses vertex r_0, r_1, \dots, r_n . $\mathcal{L}_{r_{0:n}}^{cs}$ is the concatenation of $m_{r_0 r_1}, m_{r_1 r_2}, \dots, m_{r_{n-1} r_n}$; see Fig. 7(b). The key point is that since the target moves with constant speed, the time $|m_{r_i r_{i+1}}|, i = 0, 1, \dots, n-1$ taken to traverse from r_i to r_{i+1} is proportional to the length of the edge connecting r_i and r_{i+1} , namely, $l(e_{r_i r_{i+1}})$. Therefore,

$$\begin{aligned} \mathcal{L}_{r_{0:n}}^{cs} &= \oplus_{i=0}^{n-1} m_{r_i r_{i+1}} \quad \text{with} \\ |m_{r_0 r_1}| : |m_{r_1 r_2}| : \dots : |m_{r_{n-1} r_n}| &= l(e_{r_0 r_1}) : l(e_{r_1 r_2}) : \dots : l(e_{r_{n-1} r_n}) \end{aligned} \quad (11)$$

Here, “ \oplus ” denotes concatenation. It can be shown [4] that a generative model for $\mathcal{L}_{r_{0:3}}^{cs}$ and $\mathcal{L}_{r_{0:4}}^{cs}$ is a context sensitive grammar. In this paper, we approximate $\mathcal{L}_{r_{0:3}}^{cs}, \mathcal{L}_{r_{0:4}}^{cs}$ by

$$\begin{aligned} \mathcal{L}_{r_{0:3}}^{cs} &= m_{r_0 r_1} m_{r_1 r_2} m_{r_2 r_3} \quad \text{with} \\ |m_{r_0 r_1}| : |m_{r_2 r_3}| &= l(e_{r_0 r_1}) : l(e_{r_2 r_3}) \end{aligned}$$

and

$$\begin{aligned} \mathcal{L}_{r_{0:4}}^{cs} &= m_{r_0 r_1} m_{r_1 r_2} m_{r_2 r_3} m_{r_3 r_4} \quad \text{with} \\ |m_{r_0 r_1}| : |m_{r_2 r_3}| &= l(e_{r_0 r_1}) : l(e_{r_2 r_3}) \end{aligned}$$

A CSCFG that generates $\mathcal{L}_{r_{0:n}}^{cs}$ ($n = 1, 2, 3, 4$) is given in Appendix C. The architecture of the mode based syntactic tracker for constant speed targets is given in Fig. 8(a).

C. Average Speed Model

Here, we describe the average speed model (denoted by $\mathcal{L}_{r_{0:n}}^{vt}$) described in Sec. II-B given a pattern $r_{0:n}$.

The average speed model $\mathcal{L}_{r_{0:n}}^{vt}$ is the sequence of vectors comprising directions and locations of a vehicular traffic that starts from r_0 , traverses through r_1, r_2, \dots, r_{n-1} and ends at r_n . From Fig. 7(b), we have

$$\mathcal{L}_{r_{0:n}}^{vt} = \oplus_{i=0}^{n-1} m_{r_i r_{i+1}} \quad (12)$$

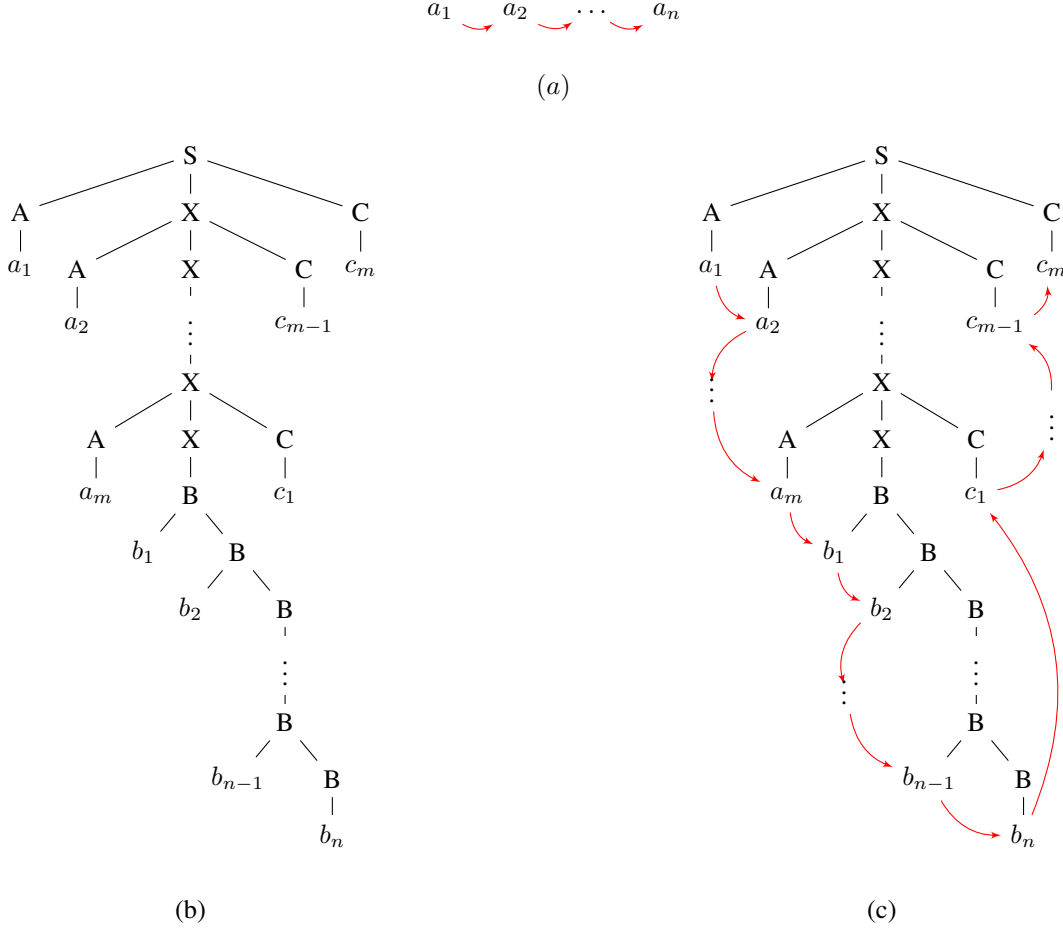


Figure 6. Comparison of dependency structures of Markov chain, SCFG and CSCFG. (a) Markov chain, (b) SCFG model, (c) CSCFG model. A CSCFG model is a combination of a Markov chain and a SCFG model. Black edges represent tree dependencies and red arrows indicate serial dependencies.

where $m_{r_i r_{i+1}}$, $i = 0, 1, \dots, n-1$ is defined in (10) and “ \oplus ” is defined in (11). There is no additional constraint on $|m_{r_i r_{i+1}}|$, $i = 0, 1, \dots, n-1$ because the speed of a vehicular traffic is location based and time varying (5). $\mathcal{L}_{r_{0:n}}^{vt}$ can be modeled via a stochastic regular grammar which is a strict subset of CSCFGs (2). The architecture of the mode based syntactic tracker for vehicular traffic is illustrated in Fig. 8(b).

IV. CSCFG-DRIVEN PARTICLE FILTER TRACKER

Thus far, we have discussed a 3-level model for a target confined to a roadmap. In this section, we describe a novel natural language based particle filtering algorithm for estimating the coordinates of the target.

For the 3-level model proposed in Sec. II and Sec. III, direct computation of the posterior distribution is computationally intractable.³ Therefore, given the noisy observation sequence $\mathbf{z}_{1:k}$, $k = 1, 2, \dots$, our aim is to compute the posterior distribution

$$p(s_k | \mathbf{z}_{1:k}, r_{0:n}^*) \quad \text{where} \quad s_k = (\mathbf{x}_k, q_k, e_k), r_{0:n}^* = \underset{\forall r_{0:n} \in M}{\operatorname{argmax}} p(r_{0:n} | \mathbf{z}_{1:k}) \quad (13)$$

³Since the model is a grammar-driven linear system, even for simple case of a jump Markov linear system, computing the posterior is intractable

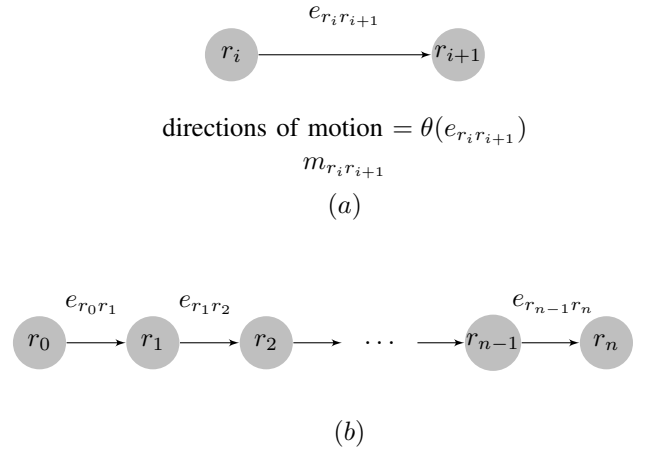


Figure 7. (a) illustrates $m_{r_i r_{i+1}}$ defined in (10). The target has a constant direction of motion $\theta(e_{r_i r_{i+1}})$ and traverses locations $r_i, e_{r_i r_{i+1}}, r_{i+1}$ in order. (b) illustrates that $\mathcal{L}_{r_{0:n}}^{cs}$ or $\mathcal{L}_{r_{0:n}}^{vt}$ is the concatenation of $m_{r_0 r_1}, m_{r_1 r_2}, \dots, m_{r_{n-1} r_n}$.

Recall from (4), (5), that e_k denotes the speed of the target at time k , \mathbf{x}_k, q_k are defined in (6), (3) and $r_{0:n}$ denotes a pattern defined in (9). In this section, a Rao-Blackwellized CSCFG-driven particle filtering algorithm is derived (see Algorithm 1

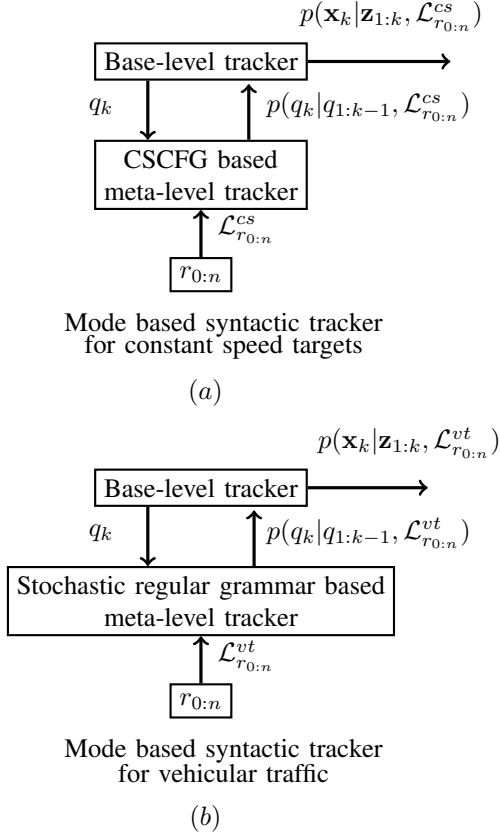


Figure 8. Architectures of the mode based syntactic trackers for two types of targets. (a) is the architecture of the mode based syntactic tracker for constant speed targets. (b) is the architecture of the mode based syntactic tracker for vehicular traffic. $r_{0:n}$ is a pattern defined in (9). $\mathcal{L}_{r_{0:n}}^{cs}$ and $\mathcal{L}_{r_{0:n}}^{vt}$ are, respectively, the constant speed model and the average speed model for pattern $r_{0:n}$. k is the discrete time, $\mathbf{z}_{1:k}$ denotes the noisy observation sequence recorded by a GMTI radar and \mathbf{x}_k is the state vector of the target. The mode q_k is defined in (3).

and Algorithm 2). The particle filtering algorithm presented below exploits the fact that given the kinematic state \mathbf{x}_k , the modes can be estimated via a finite dimensional predictor in terms of the Earley Stolcke parser.

For constant speed targets

$$p(s_k^i | \mathbf{z}_{1:k-1}, r_{0:n}) = p(s_k^i | \mathbf{z}_{1:k-1}, \mathcal{L}_{r_{0:n}}^{cs})$$

where $\mathcal{L}_{r_{0:n}}^{cs}$ is the constant speed model discussed in Sec. III-B. For the vehicular traffic,

$$p(s_k^i | \mathbf{z}_{1:k-1}, r_{0:n}) = p(s_k^i | \mathbf{z}_{1:k-1}, \mathcal{L}_{r_{0:n}}^{vt})$$

where $\mathcal{L}_{r_{0:n}}^{vt}$ is the average speed model discussed in Sec. III-C.

The particle approximation to the full posterior distribution is

$$p(s_{1:k} | \mathbf{z}_{1:k}, r_{0:n}) \approx \sum_{i=1}^{N_p} w_k^i \delta(s_{1:k}^i)$$

where N_p is the number of particles. The particle weights w_k^i

are computed recursively as

$$\begin{aligned} w_k^i &= \frac{p(s_{1:k}^i | \mathbf{z}_{1:k}, r_{0:n})}{\pi(s_{1:k}^i | \mathbf{z}_{1:k}, r_{0:n})} \\ &\propto \frac{p(s_k^i | s_{1:k-1}^i, r_{0:n}) p(\mathbf{z}_k | s_k^i)}{\pi(s_k^i | s_{1:k-1}^i, \mathbf{z}_{1:k}, r_{0:n})} \frac{p(s_{1:k-1}^i | \mathbf{z}_{1:k-1}, r_{0:n})}{\pi(s_{1:k-1}^i | \mathbf{z}_{1:k-1}, r_{0:n})} \\ &\propto \frac{p(s_k^i | s_{1:k-1}^i, r_{0:n}) p(\mathbf{z}_k | s_k^i)}{\pi(s_k^i | s_{1:k-1}^i, \mathbf{z}_{1:k}, r_{0:n})} w_{k-1}^i \end{aligned}$$

Choose the bootstrap proposal distribution

$$\begin{aligned} \pi(s_k^i | s_{1:k-1}^i, \mathbf{z}_{1:k}, r_{0:n}) &= p(s_k^i | s_{1:k-1}^i, r_{0:n}) \\ &= p(\mathbf{x}_k^i, q_k^i, e_k^i | \mathbf{x}_{1:k-1}^i, q_{1:k-1}^i, e_{1:k-1}^i, r_{0:n}) \\ &= p(q_k^i | q_{1:k-1}^i, r_{0:n}) p(e_k^i | e_{1:k-1}^i, l_k^i) p(\mathbf{x}_k^i | \mathbf{x}_{k-1}^i, d_{k-1:k}^i, e_{k-1:k}^i) \end{aligned} \quad (14)$$

The first term is the CSCFG one-step predictor in (25). The second term is computed in terms of (4) and (5). The third term in (14), according to (6) is

$$\begin{aligned} p(\mathbf{x}_k^i | \mathbf{x}_{k-1}^i, d_{k-1:k}^i, e_{k-1:k}^i) &= \mathcal{N}(f(\mathbf{x}_{k-1}^i, d_{k-1:k}^i, e_{k-1:k}^i), \tilde{Q}_k(d_{k-1:k}^i)) \end{aligned} \quad (15)$$

where $f(\cdot)$ and $\tilde{Q}_k(d_{k-1:k}^i)$ are defined in (6). Recursive computation of w_k^i before normalization is

$$\begin{aligned} \tilde{w}_k^i &= p(\mathbf{z}_k | s_k^i) w_{k-1}^i \quad \text{with} \\ p(\mathbf{z}_k | s_k^i) &= \begin{cases} 0 & l_k^i \neq B(\mathbf{x}_k^i) \\ \mathcal{N}(h(\mathbf{x}_k^i, \mathbf{c}_k), R) & l_k^i = B(\mathbf{x}_k^i) \end{cases} \end{aligned} \quad (16)$$

where $h(\cdot)$, \mathbf{c}_k , R are defined in (8) and B is defined in (6). The CSCFG-driven particle filtering algorithm is illustrated in Algorithm 1.

Algorithm 1 CSCFG-Driven Particle Filter Tracker

Function CSCFG-PF $\{s_{1:k-1}^i, w_{k-1}^i, \mathbf{z}_k, r_{0:n}\}$
for $i=1$ **to** N_p **do**
 Sample $q_k^i \sim p\{q_k | q_{1:k-1}^i, r_{0:n}\}$ using (25)
 Sample e_k^i according to (4) or (5)
 Sample \mathbf{x}_k^i according to (15)
 Compute \tilde{w}_k^i using (16)
end for
 $W_k = \sum_{i=1}^{N_p} \tilde{w}_k^i$
 Normalize $w_k^i = \tilde{w}_k^i / W_k, \forall i = 1, 2, \dots, N_p$
 $N_{\text{effective}} = \frac{1}{\sum_{i=1}^{N_p} (w_k^i)^2}$
if $N_{\text{effective}} < \text{threshold}$ **then**
 RESAMPLE
 for $i=1$ **to** N_p **do**
 $w_k^i = \frac{1}{N_p}$
 end for
end if
return $\{s_{1:k}^i, w_k^i\}$

Decision Directed Scheme

Strictly speaking, we should assign each particle a CSCFG parser. Here, we apply a decision-directed scheme introduced in [9], namely, N_p particles drive a single parser. Denote \hat{q}_k the soft estimate at time k and is computed as

$$p(\hat{q}_k = m) = \sum_{i=1}^{N_p} \delta(q_k^i - m) w_k^i \quad (17)$$

The one step predictor in (14) is approximated by

$$p(q_k^i | q_{1:k-1}^i, r_{0:n}) \approx p(q_k^i | q_{k-1}^i, \hat{q}_{1:k-2}, r_{0:n}) \quad (18)$$

where $\hat{q}_{1:k-2} = (\hat{q}_1, \hat{q}_2, \dots, \hat{q}_{k-2})$. (18) is computed via (26) in Appendix A. A CSCFG-driven particle filtering algorithm with a decision directed scheme is presented in Algorithm 2. The posterior probability $p(r_{0:n} | \mathbf{z}_{1:k})$ in (13) is computed

Algorithm 2 CSCFG-Driven Particle Filter Tracker with Decision Directed Scheme

Function CSCFG-PF $\{s_{1:k-1}^i, w_{k-1}^i, \mathbf{z}_k, \hat{q}_{1:k-2}, r_{0:n}\}$
for $i = 1$ **to** N_p **do**
 Sample $q_k^i \sim p\{q_k | q_{k-1}^i, \hat{q}_{1:k-2}, r_{0:n}\}$ using (26)
 Sample e_k^i according to (4) or (5)
 Sample \mathbf{x}_k^i according to (15)
 Compute \tilde{w}_k^i using (16)
end for
 $W_k = \sum_{i=1}^{N_p} \tilde{w}_k^i$
 Normalize $w_k^i = \tilde{w}_k^i / W_k, \forall i = 1, 2, \dots, N_p$
 $N_{\text{effective}} = \frac{1}{\sum_{i=1}^{N_p} (w_k^i)^2}$
if $N_{\text{effective}} < \text{threshold}$ **then**
 RESAMPLE
 for $i=1$ **to** N_p **do**
 $w_k^i = \frac{1}{N_p}$
 end for
end if
 Compute \hat{q}_k using (17)
return $\{s_{1:k}^i, w_k^i, \hat{q}_{1:k}\}$

using Bayes' formula,

$$p(r_{0:n} | \mathbf{z}_{1:k}) = \frac{p(\mathbf{z}_{1:k} | r_{0:n}) p(r_{0:n})}{\sum_{r_{0:n} \in M} p(\mathbf{z}_{1:k} | r_{0:n}) p(r_{0:n})}$$

Here, $p(\mathbf{z}_{1:k} | r_{0:n})$ equals the prefix probability $p(\hat{q}_{1:k} | r_{0:n})$ computed by (27). M denotes a set of patterns that a radar operator is interested in. $p(r_{0:n})$ is the prior probability for pattern $r_{0:n}$ and $\sum_{r_{0:n} \in M} p(r_{0:n}) = 1$.

V. EXAMPLES OF ANOMALOUS TRAJECTORIES MODELED BY CSCFG

In this section, we consider two detailed examples of suspicious (anomalous) trajectories of a target: (i) round trip trajectories which indicate a target circling an area of interest, and (ii) cost constrained palindrome trajectories which indicate a target re-tracing its path. In both examples, a CSCFG serves as a generative model. These examples generalize our earlier work [5] where SCFGs were used.

It is assumed that the target is moving confined to a square grid illustrated in Fig. 9 and is tracked by a GMTI radar. The aim is to compute the maximum a posterior sequence estimate

$$\mathbf{x}_{1:n}^* = \underset{\mathbf{x}_{1:n}}{\operatorname{argmax}} p(\mathbf{x}_{1:n} | \mathbf{z}_{1:n}, \mathcal{L}) \quad (19)$$

Here, $\mathbf{x}_{1:n} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)$ denotes the target's state sequence and $\mathbf{z}_{1:n} = (\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_n)$ denotes the noisy GMTI radar measurement sequence; $\mathbf{x}_k, \mathbf{z}_k, k = 1, 2, \dots, n$ are defined later. \mathcal{L} is the round trip or cost constrained palindrome trajectory modeled by a CSCFG.

Let o_{ij} denote the node with Cartesian coordinates (i, j) and G_{grid} denote the set of all nodes of the square grid. Define the following relationships on nodes in G_{grid}

$$\begin{aligned} o_{ij} &\xrightarrow{\text{up, right}} o_{i'j'} \text{ if } i' = i, j' - j = 1 \text{ or } i' - i = 1, j' = j \\ o_{ij} &\xrightarrow{\text{down, left}} o_{i'j'} \text{ if } i = i', j' - j = -1 \text{ or } i' - i = -1, j' = j \\ o_{ij} &\xrightarrow{\text{adjacent}} o_{i'j'} \text{ if } o_{ij} \text{ and } o_{i'j'} \text{ are adjacent} \end{aligned} \quad (20)$$

At each discrete time, we assume a target can only move up, down, left, right to its adjacent nodes. \mathbf{x}_k is specified by

$$\mathbf{x}_k = \{x_k, y_k\} \in G_{\text{grid}} \quad (21)$$

\mathbf{z}_k is the two dimensional (comprising range and azimuth components) noisy observation vector recorded by a GMTI radar and the observation function is described in (8). Note that the target's position in the x (y) direction equals the coordinate in the x (y) direction times the size of the unit grid.

A. Example 1. Round Trip Model

In the round trip model, denoted by $\mathcal{L}_{\text{round}}$, a target departs from some node A , arrives at another node B and finally returns to A . On the forward trip, it is assumed that the target can only move up and right; while on the return trip, the target can only move down and left. The resulting target's trajectory is as follows:

Definition 1. $\mathcal{L}_{\text{round}} = \mathbf{x}_{1:n}$ where

$$\begin{cases} \mathbf{x}_k \xrightarrow{\text{up, right}} \mathbf{x}_{k+1} & \forall k = 1, 2, \dots, \frac{n-1}{2} \\ \mathbf{x}_k \xrightarrow{\text{down, left}} \mathbf{x}_{k+1} & \forall k = \frac{n+1}{2}, \frac{n+3}{2}, \dots, n-1 \end{cases}$$

Here, $\xrightarrow{\text{up, right}}$, $\xrightarrow{\text{down, left}}$ are defined in (20) and \mathbf{x}_k is defined in (21). An example of the round trip model is presented in Fig. 9. $\mathcal{L}_{\text{round}}$ indicates a target moves to some other node and returns to the start point with minimal distance and possibly minimal probability being detected (routine of return trip is probably different from that of forward trip). From Def. 1, $\mathcal{L}_{\text{round}}$ is the concatenation of two equal length (tree dependency) Markov chains (serial dependency): $\mathbf{x}_{1: \frac{n+1}{2}}$ and $\mathbf{x}_{\frac{n+1}{2}: n}$. The transition probability is dependent on the traffic data. Such a tree and serial combined dependency cannot be modeled via a SCFG as explained in Sec. III-A. $\mathcal{L}_{\text{round}}$ is modeled via a CSCFG and details are illustrated in Appendix B.

B. Example 2. Cost Constrained Palindrome

In a palindrome trajectory model, a target starts from some node A , moves to some node B and then retraces its path to A . Several examples in surveillance involve detecting palindrome trajectories of targets.⁴ An example of $\mathcal{L}_{\text{palindrome}}$ is shown in Fig. 9. It is well known that a SCFG is a generative model for a palindrome.

Here we consider a generalization called the cost constrained palindrome which has serial dependencies and therefore requires a CSCFG model. The serial dependencies are introduced by considering the process Δ_k which denotes the cost (threat level or fuel consumption) incurred at each time k . Δ_k is modeled by a Markov chain with state space dependent on \mathbf{x}_{k-1} and \mathbf{x}_k . Then $\sum_{i=1}^k \Delta_i$ denotes the cost (fuel or funds) incurred up to time k . We assume that the target can only continue moving if the total cost (threat) accrued lies within the bound: $\sum_{i=1}^k \Delta_i \leq C$ for some pre-specified constant C .

The cost constrained palindrome model, denoted by $\mathcal{L}_{\text{palindrome}}$ is formally specified as follows:

Definition 2. $\mathcal{L}_{\text{palindrome}} = \mathbf{y}_{1:n}$, $\mathbf{y}_k = \{\mathbf{x}_k, \Delta_k\}$ where

$$\begin{aligned} \mathbf{x}_k &\xrightarrow{\text{adjacent}} \mathbf{x}_{k+1}, k = 1, 2, \dots, n-1 \\ \mathbf{x}_n \mathbf{x}_{n-1} \dots \mathbf{x}_{\frac{n+1}{2}} &= \mathbf{x}_1 \mathbf{x}_2 \dots \mathbf{x}_{\frac{n+1}{2}} \\ \mathbf{x}_k &\sim p(\mathbf{x}_k | \mathbf{x}_{k-1}), k = 1, 2, \dots, \frac{n+1}{2} \\ \Delta_k &\sim p(\Delta_k | \Delta_{k-1}, \mathbf{x}_k, \mathbf{x}_{k-1}), k = 1, 2, \dots, n \\ \sum_{i=1}^k \Delta_i &\leq C, k = 1, 2, \dots, n \end{aligned}$$

Here, \mathbf{x}_k is defined in (21) and $\xrightarrow{\text{adjacent}}$ is defined in (20). The third item in Def. 2 is dependent on the traffic data and the fourth item specifies the evolution of the cost process.

Note that the second, third and fourth items in Def. 2 exhibit a combined tree and serial dependency which cannot be modeled by a SCFG; recall the discussion in Sec. III-A. Details of $\mathcal{L}_{\text{palindrome}}$ constructed as a CSCFG are given in Appendix B.

Finally, computing the MAP sequence estimate in (19) for a round trip or a cost constrained palindrome trajectory uses a CSCFG Viterbi tracker which is illustrated in Fig. 10. The CSCFG Viterbi parsing algorithm is detailed in Appendix B.

VI. NUMERICAL EXAMPLES

This section presents three detailed numerical examples involving CSCFGs for meta-level tracking. For the first two examples, the setup is as follows. The mode based syntactic tracker proposed in this paper estimates the state moving of a target that is confined to a roadmap given noisy GMTI radar observations. The main algorithm (Algorithm 2) is a Rao-Blackwellized particle filter which combines the Earley Stolcke parser with a base-level particle filter. The empirical

⁴Here are three examples: (i) For a smuggler, if the forward trip from A to B is safe (e.g., no guards), then retracing this path from B to A minimizes being captured. (ii) A vehicle (bus) transports passengers from A to B and on the return trip transports passengers from B to A via the same stops. (iii) Retracing the path also occurs when searching for a dropped or lost item.

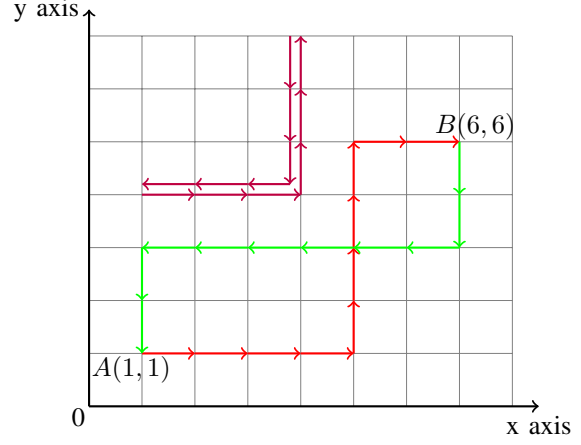


Figure 9. Examples of $\mathcal{L}_{\text{round}}$ and $\mathcal{L}_{\text{palindrome}}$ on the square grid. For $\mathcal{L}_{\text{round}}$, a target starts at node A , moves right and up to node B (red line) and moves left and down to return to A (green line). For $\mathcal{L}_{\text{palindrome}}$, a target backtracks its path (purple line).

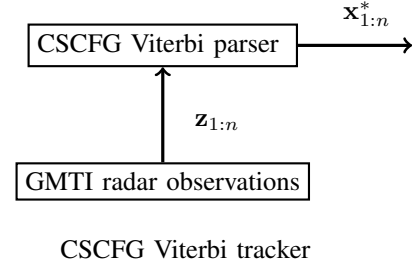


Figure 10. The architecture of the CSCFG Viterbi tracker. $\mathbf{z}_{1:n}$ is the noisy observation sequence recorded by a GMTI radar and $\mathbf{x}_{1:n}^*$ is defined in (19).

performance of the mode based syntactic tracker (denoted by M in the figures below) is evaluated by simulating measurements from a GMTI radar against a target moving on the road network described by Fig. 1. See Table I for the primary properties of the radar measurements; further detail is provided in Appendix D. Note we are assuming no missing measurements (i.e. the probability of detection is unity). The aircraft where a GMTI radar is mounted starts from (-30 km, -30 km) with (constant) altitude 3000m and moves with constant velocity (100 m/s in the x direction and 20 m/s in the y direction). We also evaluate the performance of the baseline VSIMM tracker (proposed in [1] [2]) for comparison. The performance are evaluated by the simulated sample path state root mean square error averaged over 50 independent trials. Root mean square error at time k is computed as

$$\text{RMSE}(k) = \sqrt{\frac{1}{N} \sum_{n=1}^N (\hat{\mathbf{x}}_k^n - \mathbf{x}_k^t)' (\hat{\mathbf{x}}_k^n - \mathbf{x}_k^t)}$$

Here, $\hat{\mathbf{x}}_k^n$ is the estimated target state vector at time k in the n_{th} trial and \mathbf{x}_k^t is the target's true state vector at time k .

A. Example 1. Constant Speed Targets

Consider the model of Sec. III-B. The speed of a constant speed target is an unknown constant and we assume its initial

Table I
PARAMETERS USED IN THE
SYNTACTIC TRACKING
SIMULATIONS

T	0.2s
number of particles	700
σ_{r_k}	5m
$\sigma_{\dot{r}_k}$	0.3 m/s
σ_{θ_k}	0.5°
σ_a	0.1
σ_o	0.001

σ_{r_k} , $\sigma_{\dot{r}_k}$ and σ_{θ_k} are defined in (8). T , σ_a and σ_o are defined in (6).

Table II
PERFORMANCE OF TRACKING CONSTANT SPEED TARGETS

pattern	speed		M	VSIMM	improvement
$v_3v_0v_2$	2 m/s	average	4.0064	4.7274	
		peak	4.8093	7.6005	13.87%
$v_9v_0v_3v_8$	2 m/s	average	2.8432	3.1155	
		peak	4.4128	4.7523	8.32%
$v_9v_0v_1v_4$	3 m/s	average	2.8012	3.0669	
		peak	4.3233	5.1304	6.60%
$v_5v_1v_0v_3$	3 m/s	average	3.3663	3.5585	
		peak	5.9035	8.8663	4.49%
$v_6v_2v_0v_3v_8$	4 m/s	average	3.1753	3.7875	
		peak	3.9942	5.6048	15.51%
$v_0v_2v_7v_3v_8$	4 m/s	average	3.0721	3.1331	
		peak	4.7566	4.8202	1.72%

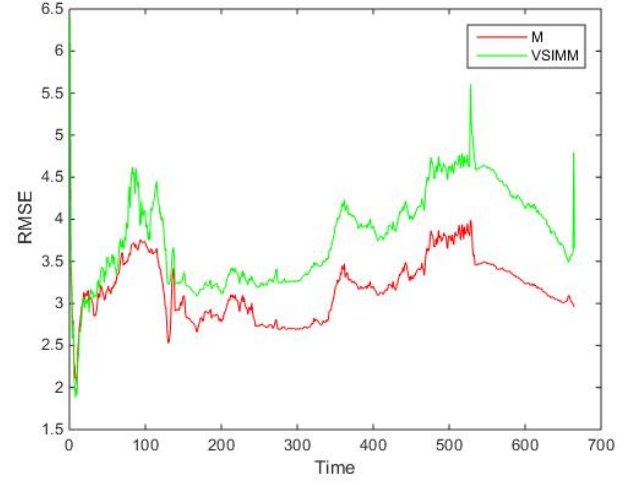
distribution is evenly distributed on 3 states: 2 m/s, 3 m/s and 4 m/s. Performance of the mode based syntactic tracker and the baseline VSIMM tracker when tracking constant speed targets under different patterns are displayed in Table II. The “improvement” in the final column is computed as

$$\text{improvement} = 1 - \text{average}\left(\frac{\text{RMSE}^M}{\text{RMSE}^{VSIMM}}\right)$$

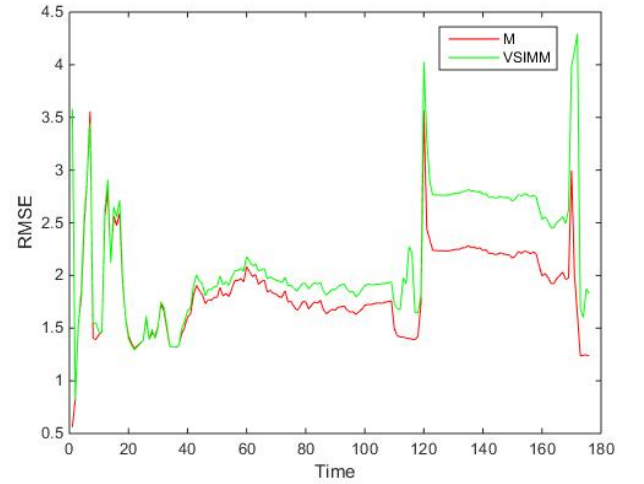
where $\text{average}\left(\frac{\text{RMSE}^M}{\text{RMSE}^{VSIMM}}\right)$ is the average ratio between the RMSE for the mode based syntactic tracker and the RMSE for the VSIMM tracker at each discrete time. An example of RMSEs for the two trackers is illustrated in Fig. 11(a). We see that the mode based syntactic tracker performs better than the VSIMM tracker in terms of the positive “improvement” which is more than 5% under 3 specific patterns: $v_3v_0v_2$, $v_9v_0v_3v_8$, $v_9v_0v_1v_4$. The explanation is that the meta-level modeling in the mode based syntactic tracker is pattern based and therefore the tracker can definitely know which road to move on to

Table III
PERFORMANCE OF TRACKING VEHICULAR TRAFFIC

pattern		M	VSIMM	improvement
$v_3v_0v_2$	average	1.8634	2.1650	
	peak	3.5637	4.2937	12.00%
$v_9v_0v_3v_8$	average	1.5997	1.7897	
	peak	5.2699	5.2661	8.45%
$v_9v_0v_1v_4$	average	2.1414	2.1823	
	peak	6.5201	7.1134	0.93%
$v_5v_1v_0v_3$	average	1.6942	1.8786	
	peak	5.6856	7.4279	2.38%
$v_6v_2v_0v_3v_8$	average	1.7125	1.7169	
	peak	8.8353	8.8324	-0.86%
$v_0v_2v_7v_3v_8$	average	2.2178	2.2004	
	peak	9.2627	8.1731	1.42%



(a)



(b)

Figure 11. Examples of RMSEs (root mean square errors) for the mode based syntactic tracker (M) and the baseline VSIMM tracker. (a) is the RMSEs when tracking a constant speed target with pattern $v_6v_2v_0v_3v_8$. (b) is the RMSEs when tracking a vehicular traffic with pattern $v_3v_0v_2$.

after passing an intersection. In addition, CSCFG at the meta-level can model longer and more complicated dependencies compared with a Markov chain such that properties of a constant speed model discussed in Sec. III-B can be well captured.

B. Example 2. Average Speed Model

Next, consider the average speed model of Sec. III-C which is suitable for vehicular traffic. For vehicular traffic, the simulations assume that the speed on roads is location-based and time-varying and is a known constant at intersections (5 m/s); see Table VI in Appendix D for details. Performance of the mode based syntactic tracker and the baseline VSIMM tracker when tracking vehicular traffic are shown in Table III. An example of RMSEs for the two trackers is illustrated in Fig. 11(b). For specific patterns such as $v_3v_0v_2$ and $v_9v_0v_3v_8$, the mode based syntactic tracker performs much better than

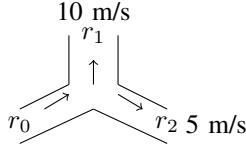


Figure 12. VSIMM tracker decreases uncertainties on which road the target is moving on by distinct speeds on different roads. r_0 , r_1 and r_2 are road names. 10 m/s and 5 m/s are the real time average speeds at some discrete time for r_1 and r_2 , respectively.

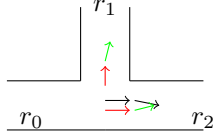


Figure 13. RMSEs for the mode based syntactic tracker and the baseline VSIMM tracker when tracking vehicular traffic decrease faster compared with those for constant speed targets. r_0 , r_1 and r_2 are road names. Δ is a positive integer. Black arrows denote target's true state vectors at time k (left) and $k + \Delta$ (right), respectively. Red arrows denote particles at time k and green arrows denote particles at time $k + \Delta$.

the VSIMM tracker by over 8%. However, there is little difference between the two competing models for the remaining 4 patterns in Table III. Generally speaking, the mode based syntactic tracker for vehicular traffic does not show as drastic an improvement as for constant speed targets. The explanation is that the VSIMM tracker is able to infer which road the target is moving on after passing an intersection by distinct average speed which is location based (5). An example is shown in Fig. 12 where a target moves from r_0 to r_2 . Due to uncertainties on which road the target moves on (r_1 or r_2) after the intersection, the VSIMM tracker propagates particles on both r_1 and r_2 . However, particles on r_1 are assigned (real time average) speed of 10 m/s which is much larger than the (real time average) speed on r_2 (5 m/s) where the target is truly moving on. This may result in obvious differences on the range rate component in the observation vector in (8). Hence, particles on r_1 have much lower weights compared with those on r_2 . In this way, (real time average) speeds for different roads give “hints” to the VSIMM tracker on which road the target is moving on after passing an intersection and thus VSIMM tracker can improve the tracking accuracy. In addition, notice that RMSEs for both trackers (when tracking vehicular traffic) decrease faster than those for constant speed targets. The reason is that vehicular traffic have higher speeds. An example is shown in Fig. 13. At time k , particles (red arrows) moving on r_1 and r_2 have similar weights. However, particles (green arrow on r_1) moving on r_1 immediately become “far away” from target's true state vector (right black arrow) after time Δ and therefore are assigned lower weights. Smaller Δ due to higher speeds of vehicular traffic results in sharp decrease in the RMSEs as shown in Fig. 11(b).

C. Example 3. Round Trip Model

The round trip model describes a target that departs from some node A , arrives at another node B and then returns to A ; see Sec. V-A. The trajectory is constrained as follows: the

target moves to its adjacent nodes at each discrete time; it moves up and right on the forward trip; on the return trip, it moves down and left. We present simulations on a 20×20 square grid roadmap with size of the unit grid as 2 meters (m). Targets are recorded by a GMTI radar with parameters listed in Table I. The GMTI radar is mounted on an aircraft. The aircraft starts from coordinates $(-300 \text{ m}, -300 \text{ m})$ with constant velocity (100 m/s in the x direction and 20 m/s in the y direction).

Given noisy GMTI radar measurements, our aim is to compute the most likely state sequence $\mathbf{x}_{1:n}^*$ defined in (19) and the architecture of the CSCFG Viterbi tracker is illustrated in Fig. 10. We also illustrate a hidden Markov model (HMM) Viterbi tracker for comparison and their performance are shown in Fig. 14 evaluated by averaging error over 40 independent trials. The error for each trial is computed as

$$\text{error} = \frac{1}{n} \sum_{k=1}^n \|\mathbf{x}_k^* - \mathbf{x}_k^t\|^2 \quad (22)$$

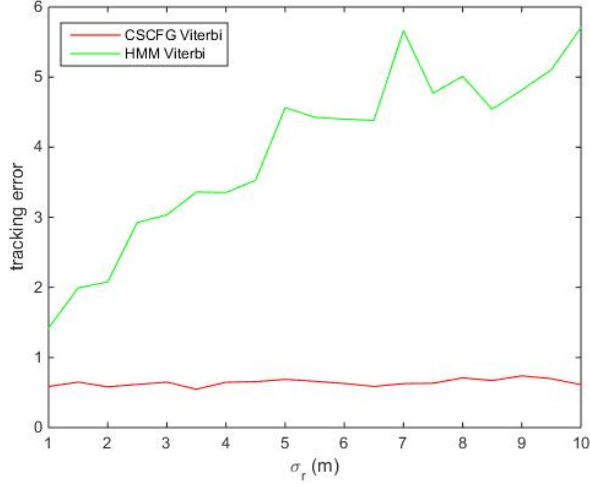
where n is the length of observations, \mathbf{x}_k^* is the Viterbi estimate for time k and \mathbf{x}_k^t is the true state vector.

Fig. 14 shows that the CSCFG Viterbi tracker has a significantly smaller tracking error (defined in (22)) compared to the HMM Viterbi tracker. The reason is that a CSCFG can model longer and more complicated spatial dependencies than a Markov chain so that all properties of a round trip model discussed in Sec. V-A are captured. Fig. 15 shows an example of state estimates by the CSCFG Viterbi tracker and the HMM Viterbi tracker. We can see for the HMM Viterbi tracker, there are lefts and downs in the forward trip (red) and rights and ups in the return trip (blue) which are impossible for the round trip model discussed in Sec. V-A.

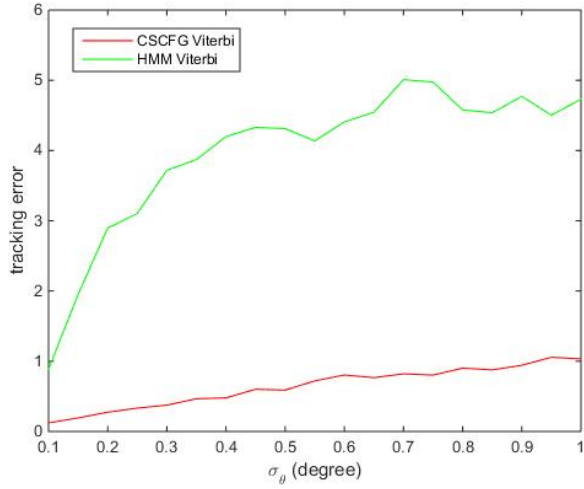
VII. CONCLUSION

In this paper, we constructed a 3-level model for the syntactic tracking problem of targets whose movement is confined to a roadmap by using natural language models. At the first level, the roadmap was modeled as a weighted, directed graph; at the second level, the mode sequence was modeled via a CSCFG; finally the base level kinematics of the target were modeled by a VSIMM. A novel CSCFG-driven particle filtering algorithm was devised to track a target's kinematics given GMTI measurements. Numerical studies show that compared with the classic VSIMM tracker, the mode based syntactic tracker proposed in this paper can substantially improve the tracking accuracy. We also discussed two CSCFG based models for targets moving on the square grid. A Viterbi algorithm was illustrated to compute the most likely hidden state sequence of the target given noisy GMTI radar measurements. Numerical results based on the round trip model show the CSCFG Viterbi tracker can substantially decrease the tracking error compared with a classical Viterbi tracker.

Acknowledgement.: We gratefully acknowledge Dr. Martie Goulding of MacDonald Dettwiler Associates (MDA) for several useful technical discussions and careful comments on the paper. Parts of this research were funded by MDA via an NSERC CRD Grant.



(a)

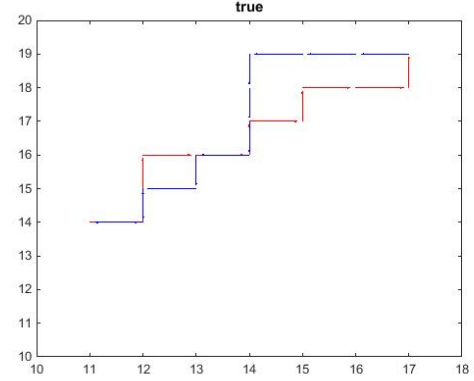


(b)

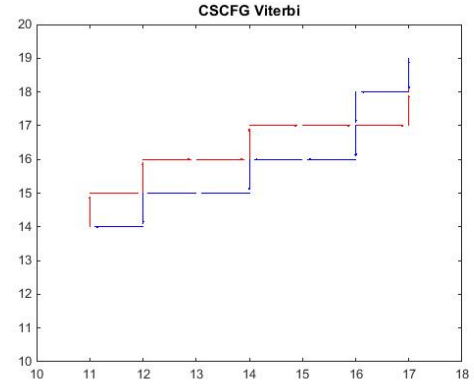
Figure 14. Performance of the CSCFG Viterbi tracker (red line) and the HMM Viterbi tracker (green line). σ_r and σ_θ are defined in (8). (a) illustrates the tracking errors for the two Viterbi trackers under a range of σ_r (step=0.5m) and we set $\sigma_\theta = 0.5^\circ$. (b) illustrates the tracking errors for the two Viterbi trackers under a range of σ_θ (step=0.05 $^\circ$) and we set $\sigma_r=5m$.

REFERENCES

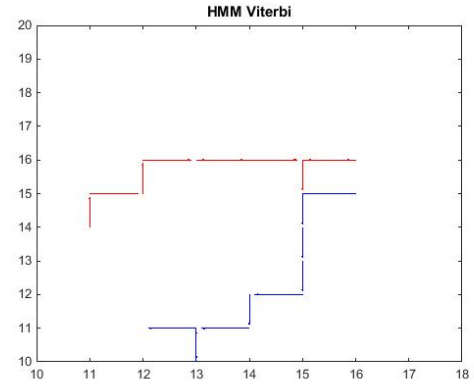
- [1] T. Kirubarajan, Y. Bar-Shalom, K. R. Pattipati, and I. Kadar, "Ground target tracking with variable structure imm estimator," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 36, pp. 26–46, Jan 2000.
- [2] H. A. P. Blom and Y. Bar-Shalom, "The interacting multiple model algorithm for systems with markovian switching coefficients," *IEEE Transactions on Automatic Control*, vol. 33, pp. 780–783, Aug 1988.
- [3] A. Wang, V. Krishnamurthy, and B. Balaji, "Intent inference and syntactic tracking with GMTI measurements," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 47, pp. 2824–2843, Oct 2011.
- [4] M. Fanaswala and V. Krishnamurthy, "Detection of anomalous trajectory patterns in target tracking via stochastic context-free grammars and reciprocal process models," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, pp. 76–90, Feb 2013.
- [5] M. Fanaswala and V. Krishnamurthy, "Spatiotemporal trajectory models for metalevel target tracking," *IEEE Aerospace and Electronic Systems Magazine*, vol. 30, pp. 16–31, Jan 2015.
- [6] K. E. Mark, M. I. Miller, and U. Grenander, "Constrained stochastic language models," in *Image Models (and Their Speech Model cousins)*, pp. 131–140, Springer, 1996.



(a)



(b)



(c)

Figure 15. Simulated performance of state estimates by the CSCFG Viterbi tracker and the HMM Viterbi tracker ($\sigma_r=5m$ and $\sigma_\theta = 0.5^\circ$). The horizontal and the vertical axis denote the x and y coordinate of the target. (a) is the true state sequence of the target. (b) displays the state estimates by the CSCFG Viterbi tracker. (c) displays the state estimates by the HMM Viterbi tracker. Red line is for the forward trip and blue for the return trip.

- [7] A. V. Aho and J. D. Ullman, *The theory of parsing, translation and compiling*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1972.
- [8] A. Stolcke, "An efficient probabilistic context-free parsing algorithm that computes prefix probabilities," *Computational linguistics*, vol. 21, no. 2, pp. 165–201, 1995.
- [9] M. Fanaswala and V. Krishnamurthy, "Syntactic models for trajectory constrained track-before-detect," *IEEE Transactions on Signal Processing*, vol. 62, pp. 6130–6142, Dec 2014.

APPENDIX A

A PARSING ALGORITHM FOR THE CONSTRAINED STOCHASTIC CONTEXT FREE GRAMMAR (CSCFG)

Here, we present a parsing algorithm (polynomial time computational cost) for the CSCFG= $\{\mathcal{N}, \mathcal{T}, S, \mathcal{R}, \mathcal{P}\}$ defined in Sec. II-B. The parsing algorithm is a modified version of the classic Earley Stolcke parser. This parsing algorithm computes the one step prediction probability and the prefix probability. To the best of our knowledge, this algorithm is new.

The classic Earley Stolcke parser is used to reconstruct parse trees that generate the input string based on a stochastic context free grammar (SCFG) [8]. It generates a sequence of Earley states by scanning, completion and prediction operations. An Earley state can be written as

$${}^k_{k'}X \rightarrow \lambda.\beta\mu[\alpha, \gamma] \quad (23)$$

where $X \in \mathcal{N}$, $\lambda, \beta, \mu \in (\mathcal{N} \cup \mathcal{T})^*$. k is the current epoch and k' is the back pointer to the epoch when this Earley state is generated by the prediction operation. The dot '.' marks that the portion on its left handed side has been parsed or recognized by the parser. α and γ are called the forward probability and the inner probability, respectively [8]. We expand the Earley state such that it can record the associated terminals

$${}^k_{k'}X \rightarrow \lambda.\beta\mu[s, f][\alpha, \gamma] \quad (24)$$

where $k, k', X, \lambda, \beta, \mu, \alpha, \gamma$ are defined in (23). s is called the *start symbol* and denotes the previous terminal when X is rewritten by $\lambda\beta\mu$ in epoch k' . f is called the *finish symbol* and denotes the final terminal that has been parsed before the dot.

The probabilistic left corner relation is computed as

$$p(X \xrightarrow{L} Y|a) = \sum_{X \rightarrow Y\lambda|a \in \mathcal{R}} p(X \rightarrow Y\lambda|a)$$

and the probabilistic unit relation is computed as

$$p(X \xrightarrow{U} Y|a) = p(X \rightarrow Y|a)$$

where $X, Y \in \mathcal{N}$, $a \in \mathcal{T}$, $\lambda \in \{\mathcal{N} \cup \mathcal{T}\}^*$. After computing the left corner and unit relations between nonterminals, we can get the reflexive, transitive left corner matrix R_l and the unit production relation matrix R_u . Details are illustrated in [8]. A parsing algorithm for the CSCFG is shown in Algorithm 3. The one step predictor is computed as

$$p(q_k|\hat{q}_{1:k-1}, \mathcal{L}_{\text{CSCFG}}) = \frac{\sum_{a=q_k, n \in u_{k-1}} \alpha({}^{k-1}_{k'}X \rightarrow \lambda.a\mu[s, f])}{\sum_{n \in u_{k-1}} \alpha({}^{k-1}_{k'}X \rightarrow \lambda.a\mu[s, f])} \quad (25)$$

where $q_k, a \in \mathcal{T}$, $X \in \mathcal{N}$, $\lambda, \mu \in (\mathcal{N} \cup \mathcal{T})^*$. $\mathcal{L}_{\text{CSCFG}}$ is the language generated by the CSCFG. $\hat{q}_{1:k-1}$ is a hard or

Algorithm 3 Parsing Algorithm for the Constrained Stochastic Context Free Grammar (CSCFG)

$X, Y, \Gamma \in \mathcal{N}$, $\lambda, \mu, \beta, \eta \in (\mathcal{N} \cup \mathcal{T})^*$, $\eta \notin \mathcal{N}$, $a \in \mathcal{T}$. n is a general denotation for an Earley state and u_k is the set of all Earley states at epoch k . \hat{q}_k is the hard or soft estimate at time k .

1. Scanning

for ${}^{k-1}_{k'}X \rightarrow \lambda.a\mu[s, f][\alpha, \gamma] \in u_{k-1}$ **do**
 Add ${}^k_{k'}X \rightarrow \lambda.a.\mu[s', f'][\alpha', \gamma']$ to u_k if $p(\hat{q}_k|a) > 0$
 $\alpha' = \alpha p(\hat{q}_k|a)$
 $\gamma' = \gamma p(\hat{q}_k|a)$
 $s' = s$
 $f' = a$

end for

$\zeta_k = \sum_{n \in u_k} \alpha({}^k_{k'}X \rightarrow \lambda.a.\mu)$

$\forall n \in u_k$, normalize α, γ using ζ_k

2. Completion

for ${}^k_{k'}\Gamma \rightarrow \eta.[s, f][\alpha, \gamma] \in u_k$ **do**

for ${}^{k'}_{k''}X \rightarrow \lambda.Y\mu[s'', s][\alpha'', \gamma''] \in u_{k'}$ **do**

if $R_u(Y, \Gamma|s) \neq 0$ **then**

Add ${}^k_{k''}X \rightarrow \lambda.Y.\mu[s', f'][\alpha', \gamma']$ to u_k

$\alpha' + = \alpha''\gamma R_u(Y, \Gamma|s)$

$\gamma' + = \gamma''\gamma R_u(Y, \Gamma|s)$

$s' = s''$

$f' = f$

end if

end for

end for

3. Prediction

for ${}^k_{k'}X \rightarrow \lambda.Y\mu[s, f][\alpha, \gamma] \in u_k$ **do**

Add ${}^k_{k'}\Gamma \rightarrow \beta.[s', f'][\alpha', \gamma']$ to u_k if $R_l(Y, \Gamma|f) \neq 0$

$\alpha' + = \alpha R_l(Y, \Gamma|f)p(\Gamma \rightarrow \beta|f)$

$\gamma' = p(\Gamma \rightarrow \beta|f)$

$s' = f$

$f' = f$

end for

soft partial observation sequence. n is a general denotation for an Earley state and u_{k-1} is the set of all Earley states at epoch $k-1$. The modified one step predictor is computed as

$$p(q_k|q_{k-1}, \hat{q}_{1:k-2}, \mathcal{L}_{\text{CSCFG}}) = \frac{\sum_{a=q_k, f=q_{k-1}, n \in u_{k-1}} \alpha({}^{k-1}_{k'}X \rightarrow \lambda.a\mu[s, f])}{\sum_{f=q_{k-1}, n \in u_{k-1}} \alpha({}^{k-1}_{k'}X \rightarrow \lambda.a\mu[s, f])} \quad (26)$$

The prefix probability is computed as

$$p(\hat{q}_{1:k}|\mathcal{L}_{\text{CSCFG}}) = \sum_{n \in u_k} \alpha({}^k_{k'}X \rightarrow \lambda.a.\mu[s, a]) \quad (27)$$

APPENDIX B

MODEL $\mathcal{L}_{\text{round}}$ AND $\mathcal{L}_{\text{palindrome}}$ VIA A CSCFG AND A VITERBI ALGORITHM FOR THE CSCFG

Here, we give constrained stochastic context free grammar models that generate $\mathcal{L}_{\text{round}}$ and $\mathcal{L}_{\text{palindrome}}$ discussed in Sec. V and introduce a Viterbi algorithm for the CSCFG.

Denote

$$\text{CSCFG}_{\text{round}} = \{\mathcal{N}, \mathcal{T}, S, \mathcal{R}, \mathcal{P}\} \quad (28)$$

the constrained stochastic context free grammar that generates $\mathcal{L}_{\text{round}}$ defined in Def. 1. $\mathcal{N} = \{S, X, A, C, D\}$ is a finite set of nonterminals and S is the start symbol. $\mathcal{T} = G_{\text{grid}}$ is a finite set of terminals where G_{grid} is the set of all nodes on the square grid discussed in Sec. V. \mathcal{R} is a finite set of rules illustrated in Fig. 16 and \mathcal{P} is a probability function on rules in \mathcal{R} . Note that the production rules in Fig. 16 include a ϵ -type production rule

$$X \rightarrow \epsilon | a, X \in \mathcal{N}, a \in \mathcal{T}$$

which indicates the nonterminal X cannot be rewritten if the previous terminal is a . The derivation stops if a ϵ -type production rule is applied. In other words, $\text{CSCFG}_{\text{round}}$ cannot generate terminal strings if ϵ -type production rules are applied in one derivation.

Denote $\text{CSCFG}_{\text{palindrome}}$ the constrained stochastic context free grammar that generates $\mathcal{L}_{\text{palindrome}}$ defined in Def. 2. The production rules are illustrated in Fig. 17.

The Viterbi algorithm is an offline algorithm to find the most likely terminal string (denoted by $a_{1:n}^*$) generated by a CSCFG given a noisy observation sequence (hard or soft) $\hat{q}_{1:n}$. The aim is to compute

$$a_{1:n}^* = \underset{a_{1:n}}{\operatorname{argmax}} p(a_{1:n} | \hat{q}_{1:n}, \mathcal{L}_{\text{CSCFG}})$$

where $\mathcal{L}_{\text{CSCFG}}$ is the language generated by the CSCFG. In the Viterbi algorithm, we extend the Earley state to be

$${}^k_{k'} X \rightarrow \lambda. \beta \mu[s, f][\gamma][str] \quad (29)$$

Definitions except $[str]$ are introduced in (24). $[str]$ is a string of clean terminals $x_{k'+1} \dots x_k$ directly or indirectly generated by the Earley state ${}^k_{k'} X \rightarrow \lambda. \beta \mu[s, f][\gamma]$. $\forall X, Y \in \mathcal{N} (X \neq Y)$, $\forall a \in \mathcal{T}$, define

$$R_u^{max}(X, Y|a) = \max p(X \rightarrow Z_1|a) p(Z_n \rightarrow Y|a) \prod_{i=1}^{n-1} p(Z_i \rightarrow Z_{i+1}|a)$$

and

$$R_u^{max}(X, X|a) = 1$$

Here, $X, Y, Z_1, Z_2 \dots Z_n$ are different nonterminals. R_u^{max} is computed to avoid completion loops. A Viterbi algorithm for the CSCFG is presented in Algorithm 4.

APPENDIX C

MODEL $\mathcal{L}_{r_{0:n}}^{cs}$ VIA A CSCFG

Here, we give production rules for the CSCFG that generates $\mathcal{L}_{r_{0:n}}^{cs}$ ($n = 1, 2, 3, 4$) defined in (11); see Fig. 18.

Algorithm 4 Viterbi Algorithm for the Constrained Stochastic Context Free Grammar (CSCFG)

$X, Y, \Gamma \in \mathcal{N}$, $\lambda, \mu, \beta \in (\mathcal{N} \cup \mathcal{T})^*$, $\beta \notin \mathcal{N}$, $a \in \mathcal{T}$. n is a general denotation for an Earley state. u_k is the set of all Earley states at epoch k . ψ_k is the set of Earley states generated by the completion operation at epoch k . \hat{q}_k is the hard or soft estimate at time k . Set $p(\hat{q}_k | \epsilon) = 0$.

1. Scanning

for ${}^{k-1}_{k'} X \rightarrow \lambda. a \mu[s, f][\gamma][str] \in u_{k-1}$ **do**
 Add ${}^k_{k'} X \rightarrow \lambda a. \mu[s', f'][\gamma'] [str']$ to u_k if $p(\hat{q}_k | a) > 0$
 $\gamma' = \gamma p(\hat{q}_k | a)$
 $s' = s$
 $f' = a$
 $str' = [str; a]$

end for

$$\zeta_k = \sum_{n \in u_k} \gamma({}^k_{k'} X \rightarrow \lambda a. \mu)$$

$\forall n \in u_k$, normalize γ using ζ_k

2. Completion

for ${}^k_{k'} \Gamma \rightarrow \beta. [s, f][\gamma][str] \in u_k$ **do**

for ${}^{k'}_{k''} X \rightarrow \lambda. Y \mu[s'', s][\gamma''] [str''] \in u_{k'}$ **do**

Add ${}^k_{k''} X \rightarrow \lambda Y. \mu[s', f'][\gamma'] [str']$ to ψ_k if
 $R_u^{max}(Y, \Gamma | s) > 0$
 $\gamma' = \gamma'' \gamma R_u^{max}(Y, \Gamma | s)$
 $s' = s''$
 $f' = f$
 $str' = [str''; str]$

end for**end for**

Add ${}^k_{k''} X \rightarrow \lambda Y. \mu[s', f'][\gamma'] [str']$ to u_k
 $\operatorname{argmax}_{\gamma \text{ to } u_k} \gamma$

3. Prediction

for ${}^k_{k'} X \rightarrow \lambda. Y \mu[s, f][\gamma][str] \in u_k$ **do**

Add ${}^k_{k'} \Gamma \rightarrow \beta. [s', f'][\gamma'] [str']$ to u_k if $R_l(Y, \Gamma | f) \neq 0$
 $\gamma' = p(\Gamma \rightarrow \beta | f)$
 $s' = f$
 $f' = f$
 $str' = \emptyset$

end for**return**

$$x_{1:n}^* = str({}_0^n S^*), p(x_{1:n}^* | \hat{q}_{1:n}, \text{CSCFG}) = \gamma({}_0^n S^*)$$

where

$${}_0^n S^* = \underset{\forall {}_0^n S \in \psi_n}{\operatorname{argmax}} \gamma$$

$\mathcal{L}_{r_{0:1}}^{cs}$ $S \rightarrow m_0 X$ $X \rightarrow m_0 X m_0$ $X \rightarrow m_1 X m_0$ $X \rightarrow m_1 X m_1$ $X \rightarrow m_2 X m_1$ $X \rightarrow m_2 m_2$ <p style="text-align: center;">(a)</p>	$\mathcal{L}_{r_{0:2}}^{cs}$ $S \rightarrow AXC$ $X \rightarrow AXC m_i \quad i = 0, 1, 2$ $X \rightarrow AC m_i \quad i = 0, 1, 2$ $A \rightarrow \underbrace{A' \dots A'}_{\tilde{n}_0} m_i \quad i = 0, 1, 2$ $A' \rightarrow m_i m_i \quad i = 0, 1, 2$ $A' \rightarrow m_{i+1} m_i \quad i = 0, 1$ $C \rightarrow \underbrace{C' \dots C'}_{\tilde{n}_1} m_i \quad i = 2, 3, 4, 5$ $C' \rightarrow m_i m_i \quad i = 3, 4, 5$ $C' \rightarrow m_{i+1} m_i \quad i = 2, 3, 4$ <p style="text-align: center;">(b)</p>
$\mathcal{L}_{r_{0:3}}^{cs}$ $S \rightarrow AXC$ $X \rightarrow AXC m_i \quad i = 0, 1, 2$ $X \rightarrow B m_2$ $A \rightarrow \underbrace{A' \dots A'}_{\tilde{n}_0} m_i \quad i = 0, 1, 2$ $A' \rightarrow m_i m_i \quad i = 0, 1, 2$ $A' \rightarrow m_{i+1} m_i \quad i = 0, 1$ $B \rightarrow m_i B m_i \quad i = 3, 4, 5$ $B \rightarrow m_{i+1} B m_i \quad i = 2, 3, 4$ $B \rightarrow m_5 m_5$ $C \rightarrow \underbrace{C' \dots C'}_{\tilde{n}_1} m_i \quad i = 5, 6, 7, 8$ $C' \rightarrow m_i m_i \quad i = 6, 7, 8$ $C' \rightarrow m_{i+1} m_i \quad i = 5, 6, 7$ <p style="text-align: center;">(c)</p>	$\mathcal{L}_{r_{0:4}}^{cs}$ $S \rightarrow AXC$ $X \rightarrow AXC m_i \quad i = 0, 1, 2$ $X \rightarrow B m_2$ $A \rightarrow \underbrace{A' \dots A'}_{\tilde{n}_0} m_i \quad i = 0, 1, 2$ $A' \rightarrow m_i m_i \quad i = 0, 1, 2$ $A' \rightarrow m_{i+1} m_i \quad i = 0, 1$ $B \rightarrow m_i B m_i \quad i = 3, 4, 5$ $B \rightarrow m_{i+1} B m_i \quad i = 2, 3, 4$ $B \rightarrow m_5 m_5$ $C \rightarrow \underbrace{C' \dots C'}_{\tilde{n}_1} m_i \quad i = 5, 6, 7, 8$ $C' \rightarrow m_i m_i \quad i = 6, 7, 8$ $C' \rightarrow m_{i+1} m_i \quad i = 5, 6, 7$ $D \rightarrow m_i D m_i \quad i = 9, 10, 11$ $D \rightarrow m_{i+1} D m_i \quad i = 8, 9, 10$ $D \rightarrow m_{11} m_{11}$ <p style="text-align: center;">(d)</p>

Figure 18. Production rules in the CSCFG that generates $\mathcal{L}_{r_{0:n}}^{cs}$ ($n = 1, 2, 3, 4$). m_0, m_1, \dots, m_{11} are defined in Table IV. \tilde{n}_0 and \tilde{n}_1 are positive integers. In (b), $\tilde{n}_0/\tilde{n}_1 = l(e_{r_0 r_1})/l(e_{r_1 r_2})$ and is irreducible. In (c), $\tilde{n}_0/\tilde{n}_1 = l(e_{r_0 r_1})/l(e_{r_2 r_3})$ and is irreducible. In (d), $\tilde{n}_0/\tilde{n}_1 = l(e_{r_0 r_1})/l(e_{r_2 r_3})$ and is irreducible.

APPENDIX D

PARAMETERS USED IN THE SIMULATIONS

Here, we give additional parameters used in our simulations. Lengths and angles of roads on roadmap in Fig. 1 are listed in Table V. Real time speeds of vehicular traffic moving on the roadmap in Fig. 1 are listed in Table VI.

Table VI
REAL TIME AVERAGE SPEEDS (m/s) OF VEHICULAR TRAFFIC ON ROADMAP IN FIG. 1

Road	Discrete Time Intervals									
	0-100	101-200	201-300	301-400	401-500	501-600	601-700	701-800	801-900	901-1000
s_0	10	14	8	10	5	12	14	10	7	10
s_1	8	10	12	5	14	7	10	10	7	13
s_2	10	7	10	14	10	5	13	14	10	10
s_3	12	7	8	10	5	8	12	10	7	10
s_4	6	13	8	10	10	10	12	10	7	10
s_5	8	13	9	9	12	7	10	10	8	10
s_6	10	13	9	9	12	7	7	10	8	10
s_7	10	13	13	12	12	7	7	10	8	10
s_8	8	10	12	7	12	12	7	12	8	12
s_9	11	10	12	7	12	12	7	12	11	12
s_{10}	11	12	10	7	10	11	8	10	10	12

$$\begin{aligned}
& \forall a_i, a_j \in \mathcal{T} \\
& S \rightarrow a_i X D_{a_i} \\
& \begin{cases} X \rightarrow AXC | a_i & a_i \neq O_{\text{northeast}} \\ X \rightarrow \epsilon | a_i & \text{otherwise} \end{cases} \\
& \begin{cases} A \rightarrow a_j | a_i & a_i \neq O_{\text{northeast}}, a_i \xrightarrow{\text{up, right}} a_j, \\ A \rightarrow \epsilon | a_i & \text{otherwise} \end{cases} \\
& \begin{cases} X \rightarrow a_j | a_i & a_i \neq O_{\text{northeast}}, a_i \xrightarrow{\text{up, right}} a_j \\ X \rightarrow \epsilon | a_i & \text{otherwise} \end{cases} \\
& \begin{cases} C \rightarrow a_j | a_i & a_i \neq O_{\text{southwest}}, a_i \xrightarrow{\text{down, left}} a_j \\ C \rightarrow \epsilon | a_i & \text{otherwise} \end{cases} \\
& \begin{cases} D_{a_i} \rightarrow a_i | a_j & a_j \xrightarrow{\text{down, left}} a_i \\ D_{a_i} \rightarrow \epsilon | a_j & \text{otherwise} \end{cases}
\end{aligned}$$

Figure 16. Production rules in the CSCFG_{round}. \mathcal{T} is defined in (28). $O_{\text{northeast}}$ and $O_{\text{southwest}}$ are the most northeast and southwest nodes on the square grid, respectively. $\xrightarrow{\text{up, right}}$ and $\xrightarrow{\text{down, left}}$ are defined in (20).

$$\begin{aligned}
& \mathbf{y}_{ki} \text{ is a terminal} \\
& S, X, Y_k \text{ are nonterminals} \\
& \forall \mathbf{y}_{ki} = \{\mathbf{x}_k, \Delta_i\} \text{ where } \mathbf{x}_k \in G_{\text{grid}}, \Delta_i \in \Delta_{\text{initial}} \\
& \text{Add } S \rightarrow \mathbf{y}_{ki} X Y_k \\
& \forall \mathbf{x}_k \xrightarrow{\text{adjacent}} \mathbf{x}_{k'} : \\
& \quad \text{Add } X \rightarrow \mathbf{y}_{k'j} X Y_{k'} | \mathbf{y}_{ki}, \quad X \rightarrow \mathbf{y}_{k'j} | \mathbf{y}_{ki} \\
& \quad \text{if } \Delta_j \in ST(\mathbf{x}_k, \mathbf{x}_{k'}) \\
& \forall \mathbf{x}_k \xrightarrow{\text{adjacent}} \mathbf{x}_{k'} : \\
& \quad \text{Add } Y_{k'} \rightarrow \mathbf{y}_{k'j} | \mathbf{y}_{ki} \\
& \quad \text{if } \Delta_j \in ST(\mathbf{x}_k, \mathbf{x}_{k'})
\end{aligned}$$

Figure 17. Production rules in the CSCFG_{palindrome}. $\xrightarrow{\text{adjacent}}$ is defined in (20). Δ_{initial} is the set of initial cost states of the target. G_{grid} is the set of all nodes in the square grid. $ST(\mathbf{x}_k, \mathbf{x}_{k'})$ is the state space of Markov chain that models cost distribution and is dependent on \mathbf{x}_k and $\mathbf{x}_{k'}$.

Table IV
SPECIFICATIONS OF $\mathcal{L}_{r_0:n}^{cs}$

parameters	direction	location
m_0		r_0
m_1	$\theta(e_{r_0 r_1})$	$e_{r_0 r_1}$
m_2		r_1
m_3		r_1
m_4	$\theta(e_{r_1 r_2})$	$e_{r_1 r_2}$
m_5		r_2
m_6		r_2
m_7	$\theta(e_{r_2 r_3})$	$e_{r_2 r_3}$
m_8		r_3
m_9		r_3
m_{10}	$\theta(e_{r_3 r_4})$	$e_{r_3 r_4}$
m_{11}		r_4

$m_i, \forall i = 0, 1, \dots, 11$ is a vector comprising directions and locations, e.g., $m_0 = \{\theta(e_{r_0 r_1}), r_0\}$.

Table V
LENGTHS AND ANGLES OF ROADS ON ROADMAP IN FIG. 1

road name	length/m	angle/radian
s_0	100	$3\pi/4$
s_1	100	$\pi/2$
s_2	200	0
s_3	100	π
s_4	100	$\pi/2$
s_5	100	$5\pi/8$
s_6	200	0
s_7	100	$-\pi/2$
s_8	100	$-\pi/2$
s_9	200	π
s_{10}	100	$\pi/2$